

LV204-01

INTER-MEMORY DATA TRANSFER CONTROLLER

Patent Number: JP9146877
 Publication date: 1997-06-06
 Inventor(s): KATO ISAO
 Applicant(s): OKI ELECTRIC IND CO LTD
 Requested Patent: JP9146877
 Application Number: JP19950328109 19951122
 Priority Number(s):
 IPC Classification: G06F13/28 ; G06F12/04
 EC Classification:
 Equivalents:

Abstract

PROBLEM TO BE SOLVED: To improve the efficiency of transfer between memories even when the addresses of the memories are misaligned.

SOLUTION: Two read buffers 41 and 42 are provided which store data of single-time access units of a source-side memory. If the address of the source-side memory is misaligned, a transfer control part 44 stores the misaligned part in one read buffer 41 and a next aligned part in the other read buffer 42. Further, the transfer control part 44 recomposes the data in the read buffers 41 and 42 so that the address is aligned with the address of a destination-side memory, and stores the data in a write buffer 43, thereby performing writing to the destination-side memory.

Data supplied from the esp@cenet database - 12

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-146877

(43) 公開日 平成9年(1997)6月6日

(51) Int.Cl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 13/28	3 1 0		G 0 6 F 13/28	3 1 0 L
12/04	5 4 0		12/04	5 4 0 A

審査請求 未請求 請求項の数 1 F D (全 5 頁)

(21) 出願番号 特願平7-328109

(22) 出願日 平成7年(1995)11月22日

(71) 出願人 000000295

沖電気工業株式会社

東京都港区虎ノ門1丁目7番12号

(72) 発明者 加藤 勲

東京都港区虎ノ門1丁目7番12号 沖電気
工業株式会社内

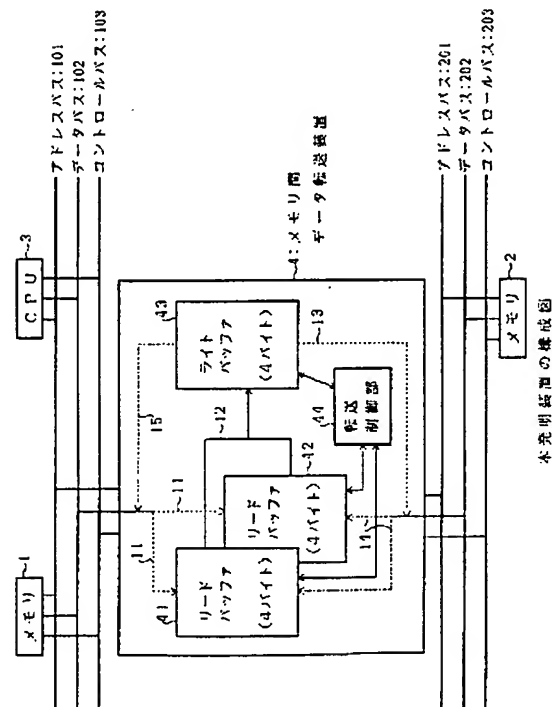
(74) 代理人 弁理士 佐藤 幸男 (外1名)

(54) 【発明の名称】 メモリ間データ転送制御装置

(57) 【要約】

【課題】 メモリのアドレスがミスアラインしている場合でもメモリ間の転送効率を向上させる。

【解決手段】 ソース側メモリの一回のアクセス単位の水データを格納するためのリードバッファ41、42を2面設ける。転送制御部44は、ソース側メモリのアドレスがミスアラインしている場合、ミスアラインしている部分を一方のリードバッファ41に、次のアラインしている部分を他方のリードバッファ42に格納する。また、転送制御部44は、デスティネーション側メモリのアドレスのアラインに一致するよう、リードバッファ41、42の水データを組み替え、そのデータをライトバッファ43に格納して、デスティネーション側メモリへのライトを行う。



【特許請求の範囲】

【請求項1】 ソース側メモリとデスティネーション側メモリ間のダイレクト・メモリ・アクセス制御を行うメモリ間データ転送制御装置において、前記ソース側メモリに対する一回のリードアクセス単位データをそれぞれ格納するための二つのリードバッファと、前記デスティネーション側メモリへの一回のライトアクセス単位データを格納するためのライトバッファと、前記二つのリードバッファに対して、前記ソース側メモリからリードアドレスのアラインに一致した単位でそれぞれ異なるデータを格納すると共に、前記デスティネーション側メモリへのライトアドレスのアラインに一致するように、前記二つのリードバッファのデータに対して組み替えを行って、前記ライトバッファに格納する転送制御部とを備えたことを特徴とするメモリ間データ転送制御装置。

【発明の詳細な説明】**【0001】**

【発明の属する技術分野】 本発明は、コンピュータシステムにおけるメモリ間のデータ転送制御を行うメモリ間データ転送制御装置に関する。

【0002】

【従来の技術】 従来のコンピュータシステムにおけるメモリ間データ転送においては、次のように構成されていた。例えば、32ビットデータバスを有するコンピュータシステムでは、4バイト単位で転送が実施され、転送元となるメモリ（以下、ソース側メモリという）のデータを4バイト分格納するレジスタ（以下、単にレジスタという）を用いてソース側メモリからデータをリードし、4バイト分のデータをレジスタに格納した後に、転送先となるメモリ（以下、デスティネーション側メモリという）に対してデータをライトすることにより実施していた。

【0003】

【発明が解決しようとする課題】 しかしながら、上記従来のメモリ転送では、4バイトが転送単位であるため、データをソース側メモリからリード、または、デスティネーション側メモリにライトする際に、メモリアドレスが4バイトでアラインしていれば、リードおよびライトサイクル共に、1回で実行されるが、ミスアラインの場合は、2回必要となり、転送時間がかかるといった問題点があった。

【0004】 このような点から、ミスアラインしている場合でもメモリ間の転送効率を向上させることのできるメモリ間データ転送制御装置の実現が望まれていた。

【0005】

【課題を解決するための手段】 本発明は、前述の課題を解決するため次の構成を採用する。

〈請求項1の構成〉 本発明のメモリ間データ転送制御装

置は、ソース側メモリとデスティネーション側メモリ間のダイレクト・メモリ・アクセス制御を行うメモリ間データ転送制御装置において、ソース側メモリに対する一回のリードアクセス単位データをそれぞれ格納するための二つのリードバッファと、デスティネーション側メモリへの一回のライトアクセス単位データを格納するためのライトバッファと、二つのリードバッファに対して、ソース側メモリからリードアドレスのアラインに一致した単位でそれぞれ異なるデータを格納すると共に、デスティネーション側メモリへのライトアドレスのアラインに一致するように、二つのリードバッファのデータに対して組み替えを行って、ライトバッファに格納する転送制御部とを備えたことを特徴とするものである。

【0006】 〈請求項1の説明〉 ソース側メモリのリードアドレスがミスアラインしている場合、転送制御部は、まず、ミスアラインしている部分を、二つのリードバッファのうちの、一方のリードバッファに格納し、次のアラインしているデータを他方のリードバッファに格納する。そして、転送制御部は、デスティネーション側メモリのライトアドレスのアラインに一致するよう、二つのリードバッファのデータに対して組み替えを行い、そのデータをライトバッファに格納する。そして、このライトバッファに格納したデータを一回のアクセス単位でデスティネーション側メモリにライトする。

【0007】 その結果、転送データが、一回のアクセス単位 $\times n$ ($n=1, 2, \dots$) であった場合、リード/ライトそれぞれのメモリへのアクセス回数は、 $n+1$ 回で済むことになる。また、一回のアクセス単位データ量は、例えば4バイトといった値であるが、これ以上の値でも同様の効果を得ることができる。

【0008】

【発明の実施の形態】 以下、本発明の実施の形態を図面を用いて詳細に説明する。

〈構成〉 図1は本発明のメモリ間データ転送制御装置の具体例を示す構成図である。図示のシステムは、独立した二つのバス間のデータ転送を実施するDMAC（ダイレクト・メモリ・アクセス・コントローラ）をバス間に配し、CPU（中央処理装置）がアドレス、データ、コントロールの各バスを介して各装置の制御およびデータ処理を行うコンピュータシステムのメモリ間データ転送に関する構成を示している。

【0009】 図のシステムは、メモリ1、2、CPU（中央処理装置）4、メモリ間データ転送制御装置4からなる。メモリ1とCPU3は、アドレスバス101、データバス102、コントロールバス103に接続され、メモリ2は、アドレスバス201、データバス202、コントロールバス203に接続されている。また、メモリ間データ転送制御装置4は、これらアドレスバス101、201、データバス102、202、コントロールバス103、203に接続されている。

【0010】メモリ間データ転送制御装置4は、CPU3に代わり、メモリ間データ転送を行うダイレクト・メモリ・アクセス・コントローラ(DMAC)であり、リードバッファ41、42、ライトバッファ43、転送制御部44を備えている。リードバッファ41、42は、それぞれソース側メモリの一回のアクセス単位分のデータを格納するためのバッファであり、本具体例では、4バイト単位となっている。ライトバッファ43は、デスティネーション側メモリの一回のアクセス単位(4バイト)分のデータを格納するためのバッファである。また、転送制御部44は、ソース側メモリより4バイト分のデータを取り出して各リードバッファ41、42に格納し、また、デスティネーション側メモリのアドレスに基づき、各リードバッファ41、42のデータをバイト単位で組み替え、デスティネーション側メモリのライトアドレスのアラインに一致するよう4バイト分のデータをライトバッファ43に格納し、デスティネーション側メモリに転送する機能を有している。

【0011】〈動作〉次に、上記構成のメモリ間データ転送制御装置の動作について説明する。例えば、メモリ1(ソース側メモリ)からメモリ2(デスティネーション側メモリ)へのデータ転送を行う場合、その転送データは、データバス102を介して破線11で示すルートで、リードバッファ41、42に格納される。そして、これらのデータに対して転送制御部44がデータ組み替えを行い、実線12で示すルートでライトバッファ43に格納し、更に、破線13で示すルートでデータバス202を介してメモリ2にライトする。一方、メモリ2(ソース側メモリ)からメモリ1(デスティネーション側メモリ)へのデータ転送を行う場合は、一点鎖線14で示すルートでリードバッファ41、42に格納し、データ組み替えを行って実線12で示すルートでライトバッファ43に格納し、一点鎖線15で示すルートでメモリ1にライトされる。

【0012】図2、3は、本具体例の動作を従来と比較して示す説明図である。尚、図中、網掛けブロック部分がメモリ上のデータ転送実施部を示しており、1ブロックが1バイトとなっている。また、転送データは $4 \times n$ バイト($n=1, 2, 3, \dots$)である。

【0013】タイプaは、転送データのメモリ上のアドレスの配置がアラインしている場合であり、この場合は、本具体例と従来とは同様の動作となる。即ち、一回のアクセスで、ABCD、EFGH、…、WXYZといったように4バイト分のアクセスが行われる。

【0014】一方、タイプb~タイプdに示すように、転送データのメモリ上のアラインが4バイト単位でない場合(ミスアライン)、転送制御部44は、まず、ミスアラインしている最初のデータをリードバッファ41に格納する。次に、アラインしている4バイトのデータをリードバッファ42に格納する。例えば、タイプbの場合、

まず、ABC部をリードし、リードバッファ41に格納する。次に、DEF部をリードし、これをリードバッファ42に格納する。従って、 $4 \times n$ バイトの転送データがあった場合、本具体例では $n+1$ 回のリードで済むことになる。

【0015】これに対し、従来の方法では、タイプbの場合、まず、ABC部をリードし、次に、4バイトにするため、D部のみリードする。そして、このような動作を繰り返すため、アクセスは、 $2 \times n$ 回となる。

【0016】また、例えば、ソース側アドレスがタイプc、デスティネーション側アドレスがタイプbといった異なるタイプのアラインであった場合、転送制御部44は、リードバッファ41、42のデータの組み替えを行ってライトバッファ43に格納する。即ち、ソース側アドレスがタイプcであるため、リードバッファ41にAB部を格納、リードバッファ42にCDEF部を格納し、これらのデータから、デスティネーション側アドレスのアラインするよう、まず、ABC部をライトバッファ43に格納し、これをメモリ2にライトする。次に、リードバッファ41にGHIJ部を格納し、リードバッファ41、42のデータより、ライトバッファ43にDEFG部を格納し、デスティネーション側メモリにライトする。このように、ソース側メモリとデスティネーション側メモリのアドレスのアラインが異なっている場合でも、各メモリ1、2へのアクセスはそれぞれ $n+1$ 回で済むことになる。

【0017】〈効果〉以上のように、本具体例では、転送データのメモリ上のアラインが4バイト単位でない場合でも、リードバッファ41、42を2面設定したことにより、4バイトずつ別々のデータをリードし格納することを可能とし、かつ、ミスアラインする部分のみデータ転送を行い、バイト単位でリードバッファ41、42のデータの入れ替えを行ってライトバッファ43に、デスティネーション側メモリのライトデータを4バイト単位でアラインするよう格納することを可能としたので、転送データが $4 \times n$ バイト($n=1, 2, 3, \dots$)のとき、メモリに対してリードおよびライトするアクセス回数が $\{2n - (n+1)\} \times 2 = 2(n-1)$ 回削減することができ、データ転送効率の向上が期待できる。

【0018】例えば、転送データ32バイトで、ソース側メモリのデータ配置がタイプb、デスティネーション側メモリのデータ配置がタイプcの場合におけるデータ転送を実施すると、まず、ソース側メモリの転送データリードアクセス回数は、従来の方法では64回必要であったが、本具体例では、33回で実施することができ、その結果、本具体例では、データ転送に要するリードおよびライトアクセス回数は66回で実施することができ、従来技術の場合の128回に比べ、よりデータ転送の向上を図ることができる。

【0019】尚、上記具体例では、4バイト単位のデー

タ転送の場合を説明したが、これに限定されるものではなく、例えば64ビットバスの場合の16バイト単位のデータ転送等であっても、同様の効果を奏することができる。

【0020】

【発明の効果】以上説明したように、本発明のメモリ間データ転送制御装置によれば、リードまたはライトのアドレスがミスアラインしている場合でも、メモリ間データ転送の効率向上を図ることができる。

【図面の簡単な説明】

【図1】本発明のメモリ間データ転送制御装置の構成図

である。

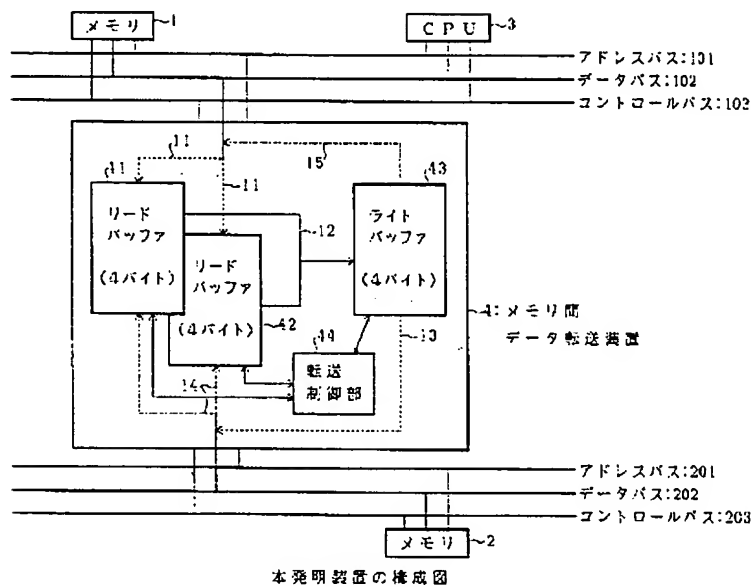
【図2】本発明のメモリ間データ転送制御装置の動作を従来と比較して示す説明図（その1）である。

【図3】本発明のメモリ間データ転送制御装置の動作を従来と比較して示す説明図（その2）である。

【符号の説明】

- 1, 2 メモリ
- 4 メモリ間データ転送制御装置
- 41, 42 リードバッファ
- 43 ライトバッファ
- 44 転送制御部

【図1】

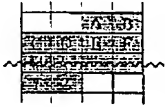
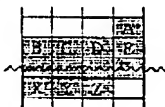


【図2】

転送データ メモリ上配置イメージ	左記転送データが ソース側アドレスの時		左記転送データが デスティネーション側アドレスの時	
	従来技術の アクセス回数	本発明による アクセス回数	従来技術の アクセス回数	本発明による アクセス回数
タイプa $m \times n$ バイト (1ブロック/1バイト)	リードアクセス n 回 ABCD部リード ... WXYZ部リード	リードアクセス n 回 ABCD部リード ... WXYZ部リード	ライトアクセス n 回 ABCD部ライト ... WXYZ部ライト	ライトアクセス n 回 ABCD部ライト ... WXYZ部ライト
タイプb 	リードアクセス $2 \times n$ 回 ABC部リード DEF部リード ... Z部リード	リードアクセス $n+1$ 回 ABC部リード DEF部リード ... Z部リード	ライトアクセス $2 \times n$ 回 ABC部ライト DEF部ライト ... Z部ライト	ライトアクセス $n+1$ 回 ABC部ライト DEF部ライト ... Z部ライト

本発明装置の動作を従来と比較して示す説明図（その1）

【図3】

転送データ メモリ上配置イメージ	左記転送データが ソース側アドレスの時		左記転送データが デスティネーション側アドレスの時	
	従来技術の アクセス回数	本発明による アクセス回数	従来技術の アクセス回数	本発明による アクセス回数
タイプc 	リードアクセス $2 \times n$ 回 AB部リード CD部リード EF部リード ⋮ YZ部リード	リードアクセス $n+1$ 回 AB部リード CDEF部リード ⋮ YZ部リード	ライトアクセス $2 \times n$ 回 AB部ライト CD部ライト EF部ライト ⋮ YZ部ライト	ライトアクセス $n+1$ 回 AB部ライト DEFG部ライト ⋮ YZ部ライト
タイプd 	リードアクセス $2 \times n$ 回 A部リード BCD部リード E部リード ⋮ XYZ部リード	リードアクセス $n+1$ 回 A部リード BCDE部リード ⋮ XYZ部リード	ライトアクセス $2 \times n$ 回 A部ライト BCD部ライト E部ライト ⋮ XYZ部ライト	ライトアクセス $n+1$ 回 A部ライト BCDE部ライト ⋮ XYZ部ライト

本発明装置の動作を従来と比較して示す説明図（その2）